SVM Kernels

COMPSCI 371D — Machine Learning

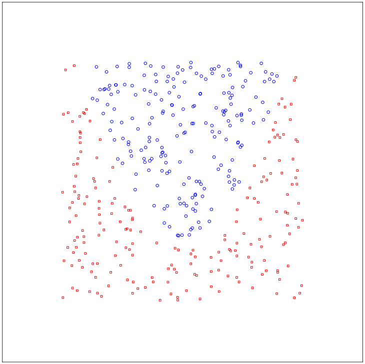
Outline

Roadmap

- SVMs so far are linear classifier, so they won't work well for non linearly separable data
- Feature augmentation: Add entries to the data point vectors $\mathbf{x}_n$ to make the data separable (or close to)
- Increases computational complexity and sample complexity (we need more training data in higher dimensions)
- The *representer theorem* lets us address this conundrum
- The effect is to make SVM decision boundaries very nonlinear
- This increases applicability of SVM enormously

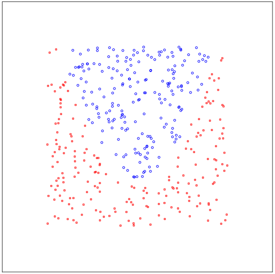
Data Representations

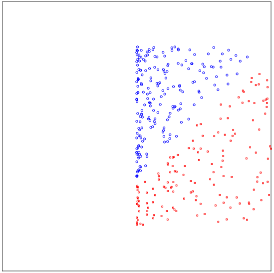
- Linear separability is a property of the data *in a given representation*
- A set that is not linearly separable. Boundary $x_2 = x_1^2$



Feature Transformations

- $\mathbf{x} = (x_1, x_2) \;\rightarrow\; \mathbf{z} = (z_1, z_2) = (x_1^2, x_2)$





- Now it is! Boundary $z_2 = z_1$

Feature Augmentation

- Feature transformation:
  $\mathbf{x} = (x_1, x_2) \ \rightarrow \ \mathbf{z} = (z_1, z_2) = (x_1^2, x_2)$
- Problem: We don't know the boundary!
- We cannot guess the correct transformation
- Feature *augmentation*:
  $\mathbf{x} = (x_1, x_2) \ \rightarrow \ \mathbf{z} = (z_1, z_2, z_3) = (x_1, x_2, x_1^2)$
- Why is this better?
- Add *many* features in the hope that some combination will help

Not Really Just a Hope!

- Add all monomials of x_1, x_2 up to some degree k
- Example: $k = 3 \ \Rightarrow \ d' = \binom{d+k}{d} = \binom{2+3}{2} = 10$ monomials
  $\mathbf{z} = (1\ ,\ \ x_1\ ,\ \ x_2\ ,\ \ x_1^2\ ,\ \ x_1 x_2\ ,\ \ x_2^2\ ,\ \ x_1^3\ ,\ \ x_1^2 x_2\ ,\ \ x_1 x_2^2\ ,\ \ x_2^3)$
- From Taylor's theorem, we know that with k high enough we can approximate *any* hypersurface by a linear combination of the features in $\mathbf{z}$
- Issue 1: Sample complexity: More dimensions, more training data (remember the curse)
- Issue 2: Computational complexity: More features, more work
- With SVMs, we can address both issues

Sample Complexity from 30,000 Feet

- The more training samples we have, the better we generalize
- With a larger N, the set T represents the *model* $p(\mathbf{x}, y)$ better
- *Sample complexity* is a measure of how many training samples (N) are needed to achieve some level of performance (error rate)
- The sample complexity of a machine learning problem turns out to grow with the dimensionality d of the data space X
- It also grows as the target error rate decreases

Sample Complexity for SVMs

- For a binary logistic-regression classifier, and given some target level of performance (error rate), the sample complexity grows linearly with the dimensionality d of X
- Not too bad, this is why linear classifiers are so successful
- SVMs *with bounded data space* X do even better
- "Bounded:" Contained in a hypersphere of finite radius
- For SVMs with bounded X, *the sample complexity is independent of* d. No curse!
- *We can augment features to our heart's content*

What About Computational Complexity?

- Remember our plan: Go from $\mathbf{x} = (x_1, x_2)$ to $\mathbf{z} = (1\,,\ \ x_1\,,\ \ x_2\,,\ \ x_1^2\,,\ \ x_1x_2\,,\ \ x_2^2\,,\ \ x_1^3\,,\ \ x_1^2x_2\,,\ \ x_1x_2^2\,,\ \ x_2^3)$ in order to make the data separable
- Can we do this without paying the computational cost?
- Yes, with SVMs

Support Vector Machines Summary

$$\hat{y} = h(\mathbf{x}) = \text{sign}(\mathbf{w}^{*T}\mathbf{x} + b^*)$$
$$b^*, \mathbf{w}^* \in \arg\min_{b,\mathbf{w}} L_T(\mathbf{w}, b)$$
$$L_T(\mathbf{w}, b) \stackrel{\text{def}}{=} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C_0}{N}\sum_{n=1}^{N} \max\{0, 1 - y_n(\mathbf{w}^T\mathbf{x}_n + b)\}$$

The Representer Theorem and Support Vectors

- The *representer theorem*: $\mathbf{w}^* = \sum_{n=1}^{N} \beta_n \mathbf{x}_n$
- *The separating-hyperplane parameter* $\mathbf{w}$ *is a linear combination of the training data points* $\mathbf{x}_n \in X \subseteq \mathbb{R}^d$
- This is surprising, especially when $N \ll d$
- It turns out that only few of the β_n are nonzero
- The corresponding data points $\mathbf{x}_n$ are called the *support vectors*
- These facts have important repercussions, so we will prove them first
  - Prove the representer theorem
  - Show why many β_n are zero

A More General Version of the Representer Theorem

- The theorem still holds if we generalize

$$L_T(\mathbf{w}, b) \;=\; \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C_0}{N}\sum_{n=1}^{N}\max\{0, 1 - y_n(\mathbf{w}^T\mathbf{x}_n + b)\}$$

to

$$L(\mathbf{w}, b) \;=\; R(\|\mathbf{w}\|) + S\left(\mathbf{w}^T\mathbf{x}_1 + b,\; \ldots,\; \mathbf{w}^T\mathbf{x}_N + b\right)$$

where

- $R(\cdot)$ is any strictly increasing function $\mathbb{R}^+ \to \mathbb{R}$
- $S(a_1, \ldots, a_N)$ is any function $\mathbb{R}^N \to \mathbb{R}$

Proof of the Representer Theorem

- If $L(\mathbf{w}, b) \;=\; R(\|\mathbf{w}\|) + S\left(\mathbf{w}^T\mathbf{x}_1 + b,\;\ldots,\;\mathbf{w}^T\mathbf{x}_N + b\right)$ where $R(\cdot)$ is strictly increasing, then $\mathbf{w}^*$ in $b^*, \mathbf{w}^* = \arg\min_{b,\mathbf{w}} L_T(\mathbf{w}, b)$ satisfies

$$\mathbf{w}^* = \sum_{n=1}^{N} \beta_n \mathbf{x}_n$$

- Restate: If

$$\mathbf{w}^* = \sum_{n=1}^{N} \beta_n \mathbf{x}_n + \mathbf{u}$$

where $\mathcal{X} = \text{span}(\mathbf{x}_1, \ldots, \mathbf{x}_N)$ and $\mathbf{u} \in \mathcal{X}^\perp$, then $\mathbf{u} = \mathbf{0}$

Proof of the Representer Theorem, Cont'd

$$[L(\mathbf{w}, b) \;=\; R(\|\mathbf{w}\|) + S\left(\mathbf{w}^T\mathbf{x}_1 + b,\; \ldots,\; \mathbf{w}^T\mathbf{x}_N + b\right)]$$

- If

$$\mathbf{w}^* = \overbrace{\sum_{n=1}^{N} \beta_n \mathbf{x}_n}^{\mathbf{w}} + \mathbf{u}$$

  where $\mathcal{X} = \text{span}(\mathbf{x}_1, \ldots, \mathbf{x}_N)$ and $\mathbf{u} \in \mathcal{X}^\perp$, then $\mathbf{u} = \mathbf{0}$
- By contradiction, assume $\mathbf{u} \neq \mathbf{0}$
- Pythagoras: $\mathbf{w} \perp \mathbf{u} \;\Rightarrow\; \|\mathbf{w}^*\|^2 = \|\mathbf{w}\|^2 + \|\mathbf{u}\|^2$
- $\mathbf{u} \neq \mathbf{0} \;\Rightarrow\; \|\mathbf{w}\| < \|\mathbf{w}^*\|$
- $R(\cdot)$ increasing $\Rightarrow\; R(\|\mathbf{w}\|) < R(\|\mathbf{w}^*\|)$

Proof of the Representer Theorem, Cont'd

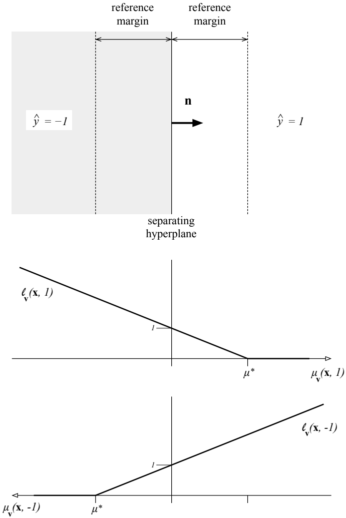
$$[L(\mathbf{w}, b) \;=\; R(\|\mathbf{w}\|) + S\left(\mathbf{w}^T\mathbf{x}_1 + b,\; \ldots,\; \mathbf{w}^T\mathbf{x}_N + b\right)]$$

- So far: $\mathbf{u} \neq \mathbf{0} \;\Rightarrow\; R(\|\mathbf{w}\|) < R(\|\mathbf{w}^*\|)$
- Since $\mathbf{u} \perp \mathbf{x}_n$, we have $\mathbf{w}^T\mathbf{x}_n + b = (\mathbf{w}^* - \mathbf{u})^T\mathbf{x}_n + b = (\mathbf{w}^*)^T\mathbf{x}_n - \mathbf{u}^T\mathbf{x}_n + b = (\mathbf{w}^*)^T\mathbf{x}_n + b$ so that $S\left(\mathbf{w}^T\mathbf{x}_1 + b,\; \ldots,\; \mathbf{w}^T\mathbf{x}_N + b\right) = S\left((\mathbf{w}^*)^T\mathbf{x}_1 + b,\; \ldots,\; (\mathbf{w}^*)^T\mathbf{x}_N + b\right)$
- Therefore, $R(\|\mathbf{w}\|) + S\left(\mathbf{w}^T\mathbf{x}_1 + b,\; \ldots,\; \mathbf{w}^T\mathbf{x}_N + b\right) < R(\|\mathbf{w}^*\|) + S\left((\mathbf{w}^*)^T\mathbf{x}_1 + b,\; \ldots,\; (\mathbf{w}^*)^T\mathbf{x}_N + b\right)$
  i.e., $L(\mathbf{w}, b) < L(\mathbf{w}^*, b)$ for all $b \Rightarrow L(\mathbf{w}, b^*) < L(\mathbf{w}^*, b^*)$
- Contradiction: $\mathbf{w}^*$ is not optimum
- Therefore $\mathbf{u} = \mathbf{0}$ and $\mathbf{w}^* = \sum_{n=1}^{N} \beta_n \mathbf{x}_n$

Many β_n are Zero

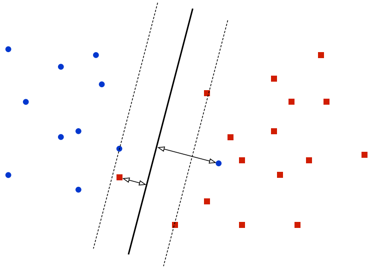
$$\mathbf{w}^* = \sum_{n=1}^N \beta_n \mathbf{x}_n$$

- $b^*, \mathbf{w}^*$ minimize the average hinge loss
- Samples that are classified correctly with margin greater than μ^* incur zero loss
- The residual risk $L(\mathbf{w}^*, b^*)$ does not depend on these samples
- Therefore $b^*, \mathbf{w}^*$ do not depend on them either
- Only samples that are either misclassified or correctly classified but with margin $\leq \mu^*$ can be in $\mathbf{w}^*$



The Support Vectors

- Only samples that are either misclassified or correctly classified but with margin less than μ^* can appear in $\mathbf{w}^*$
- These data points are called the *support vectors*



- *Sparsity:* $\mathbf{w}^* = \sum_{n \in SV} \beta_n \mathbf{x}_n$

The Sign of the Nonzero β_n

- With much heavier machinery (duality theory) it can be proven that *the sign of the nonzero β_n is y_n*:

$$\beta_n = y_n|\beta_n|$$

  (Of course the equation holds also when $\beta_n = 0$)
- We omit the proof in this course
- There may be simpler proofs, I just couldn't find one
- If you come up with one let me know!

Consequences of the Representer Theorem

- Insights from support vectors
  - Support vector machines are “more interpretable” than logistic regression classifiers
- The kernel idea
  - Feature augmentation without the computational cost

SVMs and the Representer Theorem

- Recall the formulation of SVMs
- Augment $\mathbf{x} \in \mathbb{R}^d$ to $\varphi(\mathbf{x}) \in \mathbb{R}^{d'}$, with $d' \gg d$ (typically)
- Optimal risk $L_T(\mathbf{w}^*, b^*) = \frac{1}{2}\|\mathbf{w}^*\|^2 + \frac{C_0}{N}\sum_{n=1}^N \max\{0, 1 - y_n((\mathbf{w}^*)^T\varphi(\mathbf{x}_n) + b^*)\}$
- Do inference by computing $\hat{y} = h(\mathbf{x}) = \text{sign}(\mathbf{w}^{*T}\varphi(\mathbf{x}) + b^*)$
- Plug in representer theorem: $\mathbf{w}^* = \sum_{n=1}^N \beta_n \varphi(\mathbf{x}_n)$

  $$L_T(\mathbf{w}^*, b^*) = \frac{1}{2}\|\mathbf{w}^*\|^2 + \frac{C_0}{N}\sum_{n=1}^N \max\left\{0, 1 - y_n\left(\sum_{m=1}^N \beta_m \varphi(\mathbf{x}_m)^T\varphi(\mathbf{x}_n) + b^*\right)\right\}$$

  $$\hat{y} = h(\mathbf{x}) = \text{sign}\left(\sum_{n=1}^N \beta_n \varphi(\mathbf{x}_n)^T\varphi(\mathbf{x}) + b^*\right)$$
- Data points always show up in inner products, never alone

The Kernel

$$L_T(\mathbf{w}^*, b^*) = \frac{1}{2}\|\mathbf{w}^*\|^2 + \frac{C_0}{N}\sum_{n=1}^N \max\left\{0, 1 - y_n\left(\sum_{m=1}^N \beta_m \varphi(\mathbf{x}_m)^T\varphi(\mathbf{x}_n) + b^*\right)\right\}$$

$$\hat{y} = h(\mathbf{x}) = \text{sign}\left(\sum_{n=1}^N \beta_n \varphi(\mathbf{x}_n)^T\varphi(\mathbf{x}) + b^*\right)$$

- Data points always show up in inner products, never alone
- The value $K(\mathbf{x}_m, \mathbf{x}_n) \stackrel{\text{def}}{=} \varphi(\mathbf{x}_m)^T\varphi(\mathbf{x}_n)$ is a *number*
- The optimization algorithm needs to know only $K(\mathbf{x}_m, \mathbf{x}_n)$, not $\varphi(\mathbf{x}_n)$. K is called a *kernel*. Rewrite:

$$L_T(\mathbf{w}^*, b^*) = \frac{1}{2}\|\mathbf{w}^*\|^2 + \frac{C_0}{N}\sum_{n=1}^N \max\left\{0, 1 - y_n\left(\sum_{m=1}^N \beta_m K(\mathbf{x}_m, \mathbf{x}_n) + b^*\right)\right\}$$

$$\hat{y} = h(\mathbf{x}) = \text{sign}\left(\sum_{n=1}^N \beta_n K(\mathbf{x}_n, \mathbf{x}) + b^*\right)$$

Kernel Idea 1 (Minor)

- Start with some $\varphi(\mathbf{x})$ and use the kernel to save computation
- Example: $\varphi(\mathbf{x}) =$
  $(1\,,\ x_1\,,\ x_2\,,\ x_1^2\,,\ x_1x_2\,,\ x_2^2\,,\ x_1^3\,,\ x_1^2x_2\,,\ x_1x_2^2\,,\ x_2^3)$
- Don't know how to simplify. Try this: $\varphi(\mathbf{x}) =$
  $(1\,, \sqrt{3}x_1\,, \sqrt{3}x_2\,, \sqrt{3}x_1^2\,, \sqrt{6}x_1x_2\,, \sqrt{3}x_2^2\,, x_1^3\,, \sqrt{3}x_1^2x_2\,, \sqrt{3}x_1x_2^2\,, x_2^3)$
- Can show (see notes) that
  $K(\mathbf{x}, \mathbf{x}') = \varphi(\mathbf{x})^T\varphi(\mathbf{x}') = (\mathbf{x}^T\mathbf{x}' + 1)^3$
- Something similar works for any d and k
- 4 products and 2 sums instead of 10 products and 9 sums
- Meager savings, but grows exponentially with d and k, as we know

Kernel Idea 2 (Major!)

- Just come up with $K(\mathbf{x}, \mathbf{x}')$ without knowing the corresponding $\varphi(\mathbf{x})$
- Not just *any* K. Must behave like an inner product
- For instance, $\mathbf{x}^T\mathbf{x}' = (\mathbf{x}')^T\mathbf{x}$ and $(\mathbf{x}^T\mathbf{x}')^2 \leq \|\mathbf{x}\|^2 \, \|\mathbf{x}'\|^2$ (symmetry and Cauchy-Schwartz), so we need at least
$K(\mathbf{x}, \mathbf{x}') = K(\mathbf{x}', \mathbf{x}) \quad \text{and} \quad K^2(\mathbf{x}, \mathbf{x}') \leq K(\mathbf{x}, \mathbf{x}) \, K(\mathbf{x}', \mathbf{x}')$
- These conditions are necessary, but they are not sufficient
- Fortunately, there is a theory for this

Mercer Conditions

- $K(\mathbf{x}, \mathbf{x}') : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a *kernel function* if there exists φ for which $K(\mathbf{x}, \mathbf{x}') = \varphi(\mathbf{x})^T \varphi(\mathbf{x}')$
- Finite case: Given $\mathbf{x}_n \in \mathbb{R}^d$ for $n = 1, \ldots, N$ (as in T), a symmetric function $K(\mathbf{x}, \mathbf{x}')$ is a kernel function on that set iff the $N \times N$ matrix $A = [K(\mathbf{x}_i, \mathbf{x}_j)]$ is positive semi-definite
- Problem: We would like to know if $K(\mathbf{x}, \mathbf{x}')$ is a kernel for *any* T, or even for $\mathbf{x}$ we have not yet seen
- Infinite case: $K(\mathbf{x}, \mathbf{x}')$ is a kernel function iff for every $f : \mathbb{R}^d \to \mathbb{R}$ s.t. $\int_{\mathbb{R}^d} f(\mathbf{x})\, d\mathbf{x}$ is finite, $\int_{\mathbb{R}^d \times \mathbb{R}^d} K(\mathbf{x}, \mathbf{x}')\, f(\mathbf{x})\, f(\mathbf{x}')\, d\mathbf{x}\, d\mathbf{x}' \geq 0$
- Immediate extension of positive-definiteness to the continuous case

The "Kernel Trick"

- There is a theory for checking the Mercer conditions algorithmically (eigenfunctions instead of eigenvectors)
- There is a calculus for how to build new kernel functions
- A whole cottage industry tailors kernels to problems
- This is rather tricky. However, the *Gaussian kernel* is very popular

$$K(\mathbf{x}, \mathbf{x}') = e^{-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{\sigma^2}}$$

- A measure of *similarity* between $\mathbf{x}$ and $\mathbf{x}'$
- Gaussian kernels are also called *Radial Basis Functions*

Kernels and Support Vectors

- Recall: Decision rule for SVM is $h(\mathbf{x}) = \text{sign}((\mathbf{w}^*)^T\varphi(\mathbf{x}) + b)$ (in transformed space, where the SVM is linear)
- The separating hyper-plane is $(\mathbf{w}^*)^T\varphi(\mathbf{x}) + b = 0$
- From representer theorem, $\mathbf{w}^* = \sum_n \beta_n \varphi(\mathbf{x}_n)$ where the sum is over support vectors only
- Therefore the separating hyperplane is $\sum_n \beta_n \varphi(\mathbf{x}_n)^T \varphi(\mathbf{x}) + b = 0$
- That is, $\sum_n \beta_n K(\mathbf{x}_n, \mathbf{x}) + b = 0$
- $\mathbf{x}_n$ and $\mathbf{x}$ are in the *original* data space X
- This equation describes the decision boundary induced in the *original* data space X
- An affine boundary in $\varphi(X)$ is a nonlinear boundary in X

The "Kernel Trick:" Summary, Part 1

- In a linear SVM, feature vectors $\mathbf{x}$ always show up in inner products: $\mathbf{x}_m^T\mathbf{x}_n$ or $\mathbf{x}_n^T\mathbf{x}$
- If features are augmented, $\mathbf{x} \rightarrow \varphi(\mathbf{x})$, also $\varphi(\mathbf{x})$ always shows up in inner products: $\varphi(\mathbf{x}_m)^T\varphi(\mathbf{x}_n)$ or $\varphi(\mathbf{x}_n)^T\varphi(\mathbf{x})$
- Define a kernel $K(\mathbf{x}, \mathbf{x}')$ such that there exists an (often unknown) mapping $\varphi()$ for which

$$K(\mathbf{x}, \mathbf{x}') = \varphi(\mathbf{x})^T\varphi(\mathbf{x}')$$

- We always work with $K(\mathbf{x}, \mathbf{x}')$ without ever involving $\varphi(\mathbf{x})$ or $\varphi(\mathbf{x}')$ (which are large, possibly infinite)
- We avoid the computational cost of feature augmentation

The "Kernel Trick:" Summary, Part 2

- Given $K(\mathbf{x}, \mathbf{x}')$ to there exists a mapping $\varphi()$ for which

  $$K(\mathbf{x}, \mathbf{x}') = \varphi(\mathbf{x})^T \varphi(\mathbf{x}')$$

  iff K satisfies the *Mercer condition*:

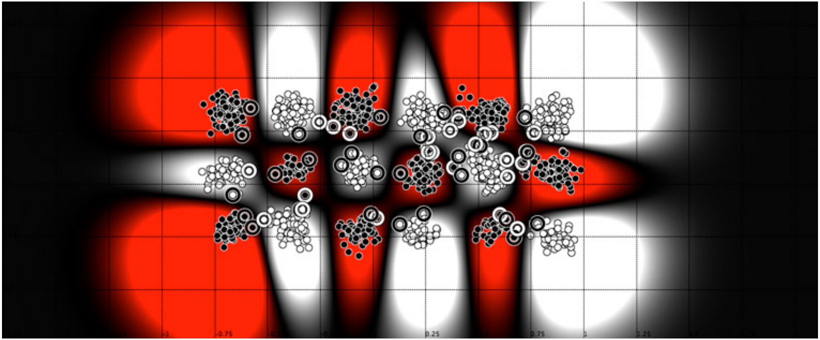
  For every $f : \mathbb{R}^d \to \mathbb{R}$ s.t. $\int_{\mathbb{R}^d} f(\mathbf{x})\, d\mathbf{x}$ is finite,

  $\int_{\mathbb{R}^d \times \mathbb{R}^d} K(\mathbf{x}, \mathbf{x}')\, f(\mathbf{x})\, f(\mathbf{x}')\, d\mathbf{x}\, d\mathbf{x}' \geq 0$
- This condition can be verified through eigenfunction computations
- Important examples: The *Radial Basis Function (RBF)* $K(\mathbf{x}, \mathbf{x}') = e^{-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{\sigma^2}}$ is a kernel
- What does the decision boundary look like now?

Gaussian Kernels and Support Vectors

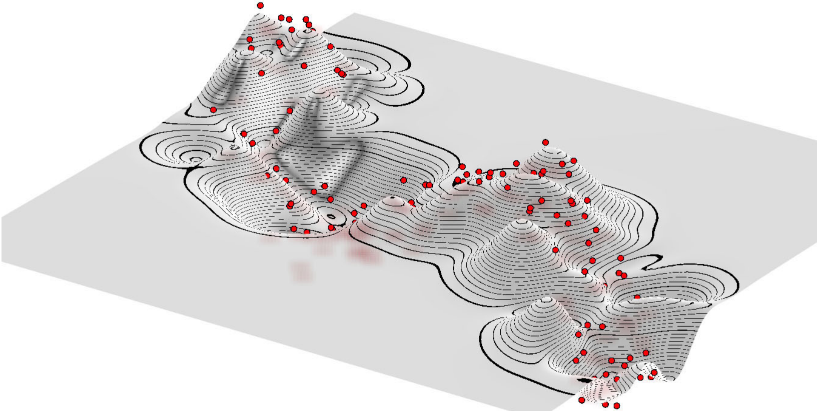
- The decision boundary *in the original space* is
  $\sum_n \beta_n K(\mathbf{x}_n, \mathbf{x}) + b = 0$
  where the sum is over support vectors
- For RBF SVMs, $\sum_n \beta_n e^{-\frac{\|\mathbf{x}-\mathbf{x}_n\|^2}{\sigma^2}} = -b$
- Simple geometric interpretation
- Recall that the sign of β_n is y_n

Classification



http://mldemos.b4silio.com

Regression



http://mldemos.b4silio.com